# 部署工作内容

## 环境部署

### a.开发环境部署

服务地址：test.q7link.com(私有IP：172.31.20.102)

代码分支：dev分支

jar包发布模式：SNAPSHOT

部署步骤：

1. 执行Jenkins任务<common\_services\_dev\_mvn/dev>，将dev分支的各个工程按SNAPSHOT模式进行发布
2. 执行Jenkins任务<noprod\_apps\_build/master>，选择相应的代码分支(dev)和环境(dev)，进行打包、生成数据库升级脚本、升级dev环境的数据库、打镜像。



3.执行Jenkins任务<all-application-deploy>，选择要部署的环境(dev)、部署的应用（后端选择apps）、打镜像的分支(第二步中选择的分支-dev）、打镜像的分支(第二步中选择的环境-dev)



### b.升级报错处理

开发/测试/hotfix/inte等各环境升级报错处理，测试使用的环境现由测试进行部署

## 代码分支管理

### a.迭代提测时

每个迭代到达提测点时，需要进行如下操作：

1. dev分支拉出release分支，并将release分支设置分支保护

（权限：merge:Maintainers push:Maintainers）

1. 将dev分支的开发脚本及相关数据修复相关接口调用信息
2. 将dev分支各各工程版本号的次版本加1，然后修改各工程依赖工程的版本号
3. 将dev分支的各工程发布

### b.迭代上线前

每个迭代当测试通知可以上线时，对release分支需要进行如下操作：

1. 将自动编译任务关闭(Jenkins: auto\_build\_job)
2. 将各工程依赖的工程版本改为RELEASE版本，然后修改build工程下的config.yaml文件，并提交。
3. 通知蒋宗良发布平台相关工程的RELEASE版本
4. 平台发布完毕之后，发布剩余工程的RELEASE版本
5. 所有工程发布完毕之后，通知测试进行部署

### c.迭代上线后

迭代上线后，需要进行如下操作：

1. 将release分支合并至master分支，如冲突，以release分支为准
2. 将release分支合并至dev分支，如冲突，找相关开发人员进行确认合并
3. 将release分支删除
4. 将自动编译任务开启(Jenkins: auto\_build\_job)

### d.hotfix/emergency工作准备

Hotfix：固定一周一次（一般为周三上线）

Emergency：线上紧急bug修复（一般为当天上线）

当接到hotfix/emergency申请时，需要进行如下操作：

1. 发起hotfix审批，hotfix由负责人进行整理统一发起，emergency由相关开发人员进行发起
2. 从master拉取hotfix/emergency分支，并将hotfix/emergency分支设置分支保护

（权限：merge:Maintainers push:No one）

1. 清空hotfix/emergency分支的开发脚本及相关数据修复相关接口调用信息
2. 将hotfix/emergency分支各工程依赖工程的版本号修改为对应工程的版本
3. 将hotfix/emergency分支的各工程发布
4. 通知开发人员进行bug修改提交

### e.hotfix/emergency上线前

参照<b.迭代上线前>

### f.hotfix/emergency上线后

hotfix/emergenc上线后，需要进行如下操作：

1. 将hotfix/emergenc分支合并至master分支，如冲突，以hotfix/emergenc分支为准
2. 将hotfix/emergenc分支合并至dev和release分支，emergency分支同时还要合并至hotfix分支。如冲突，找相关开发人员进行确认合并
3. 将hotfix/emergenc分支删除
4. 将自动编译任务开启(Jenkins: auto\_build\_job)

注：相关基础操作说明文档：<https://shimo.im/docs/rCY6VVPhQ8TkcKjg>